REMARKS

Applicant appreciates the consideration of the previously submitted arguments and notes that all anticipation and obviousness rejections presented in the prior Office Action have been abandoned. However, the current Office Action presents a new round of anticipation and obviousness rejections, based on new art ("Davenport," U.S. Pub. 2004/0263477 A1). Applicant rebuts these rejections in the below sections, and correspondingly explains the claim amendments made herein.

Davenport does not anticipate claims 1-4, 6-8, 11-14, and 16-22

The instant application as filed described Applicant's claimed subject matter in terms of computer programs and methods implemented on a personal computer (PC). In this response, Applicant amends all independent claims to stipulate computer programs executed by a PC and methods implemented on a PC. Applicant also makes corresponding dependent claim amendments as needed for language harmonization with the independent claim changes.

Davenport explicitly teaches a peripheral device external to and separate from a PC, and sets out that distinction as being a critical aspect of Davenport's invention. See paragraph [0061] of Davenport, stating that "It is a significant and salient aspect of the invention that the programmed actions and sequences are stored in non-volatile memory associated with the microcontroller 21, and thus are incorporated into the invention independently of any computer or game to which the invention 20 may be connected." The distinction that Davenport's peripheral device is separate from any associated computer or game console is reiterated throughout Davenport.

In contrast, Applicant's claimed invention is implemented in and by a PC, and all pending claims now explicitly stipulate that limitation. This distinction alone means that Davenport cannot anticipate any pending claims, because a reference anticipates only if it teaches each and every element of the claims at issue, in the identical arrangement as claimed. Davenport expressly

teaches implementation apart from a PC and calls out advantages of doing so (which appear to apply specifically to Davenport's operational context), and Davenport thus cannot anticipate any claim now pending, including any of independent claims 1, 11, 12, and 14, nor can Davenport anticipate any of their dependent claims.

Regarding claim 2 as depending from claim 1, Applicant notes the explicit limitation of claimed program instructions to create one or more operating system hooks to detect event messages associated with incoming input events corresponding to one or more types of computer input devices. The Office Action refers to paragraph [0059] of Davenport as providing anticipatory teachings, but that paragraph plainly does not support the rejection. In claim 2, "operating system" is the PC operating system, and the claim thus involves creating PC operating system hooks to detect (operating system) event messages that are associated with peripheral events incoming to the PC.

At paragraph [0059], Davenport simply states that its separate peripheral device—expressly implemented apart from any PC or game console—feeds input events to the device's microcontroller 21, which directly generates output events by mapping the input events to one of an action table or a sequence table. The thusly translated events are then passed along to a computer or game device for normal processing. These teachings self-evidently do not anticipate the expressly claimed use of PC operating system hooks and operating system event message detection. Davenport cannot be argued as anticipating claim 2, nor, likewise, as anticipating claim 3, which also includes operating system hook limitations.

Regarding claim 6 as depending from claim 1, Applicant explicitly claims program instruction to determine whether an incoming input event is swallowed or passed through. Paragraphs [0009] and [0022] of the application as filed explain, for example, that swallowing an input event prevents other event translators and/or other PC processes from "seeing" the event. Swallowing an event thus represents strategically hiding the event from further processing, either by the claimed event translators and/or by the PC operating system at large. The rejection

arguments state that paragraph [0066] of Davenport provides anticipatory teachings where inputs can be linked together if need be, which is argued as "swallowing" or "passing-through." Linking input events together cannot be legally construed as selectively swallowing an input event or passing it through.

See, e.g., paragraph [0022], where swallowing is described as:

> For example, the general action processor 26 can be configured to "swallow" or not to swallow the incoming input event. If it is configured to swallow the event, then event translator does not "pass through" the triggering input event and other programs/processes being executed by computer 8 do not see the swallowed event. In contrast, if the general action processor 26 is configured not to swallow the triggering input event, then it passes through the event, i.e., it returns the triggering event to the IDT program 10 for processing by other event translators 20, and/or for release back to the system queue for processing by the OS and other programs.

(Emphasis added.)

Self-evidently, linking inputs together does not fit the plain definition of swallowing as detailed in the application, and as would be understood by one of ordinary skill in the art, in view of the specification. Thus, Davenport does not anticipate claim 6.

Regarding claim 8, Applicant explicitly claims program instructions to determine whether an incoming input event triggers an activation of or a focus shift to a targeted program. The Office Action states that paragraphs [0070]-[0072] of Davenport provide anticipatory teachings, but that assertion is in error. The cited paragraphs of Davenport do no more than explain allegedly improved scroll wheel event processing, which Davenport refers to as "8-way" scrolling. Davenport does not teach or suggest any mechanism for determining whether an incoming input event triggers an activation of or a focus shift to a targeted program. Indeed, Davenport is explicitly implemented outside of any associated computer or game console, and does not appear to be in a position to determine whether the (translated) events it outputs for the associated computer or game console will cause a focus shift of program activation. What is clear is that Davenport itself provides no teachings that it can or does perform the claimed limitation.

Claim 15 is not obvious over Davenport

Claim 15 depends from claim 14, which is not anticipated by Davenport, and claim 15 is not made obvious by Davenport. Claim 15 stipulates that the claimed PC program of claim 14 is a WINDOWS-based program for execution on a Windows-based PC.

The obviousness rejection of claim 15 actually is unclear, because it states only that Davenport is described as interacting with a PC and that "official notice" is taken that WINDOWS is a notoriously well known PC operating system. Indeed, Davenport does describe its peripheral input system as being suitable for attaching to a PC, for use in input translation processing, and WINDOWS is a notoriously well known as a PC operating system. However, those arguments miss the question, which is whether it would be obvious for Davenport to be configured as a WINDOWS-based program for execution on a WINDOWS-based PC.

The answer is self-evidently "no." Davenport goes to great pains to explain that it is a peripheral input system for use with a PC, game console, or other digital system, but that its virtues and advantages come from being implemented separately from any associated computer system—e.g., portability between computer systems, storage of translation configurations within its own non-volatile memory, etc. If read properly, the Patent Office is saying with respect to Applicant's claim 15 that it would have been obvious to implement Davenport as a WINDOWS-based computer program on a WINDOWS-based PC for attachment to another WINDOWS-based PC, for the purpose of translating input events for that other WINDOWS-based PC. No one skilled in the art would understand that arrangement as obvious.

Claims 9 and 10 are not obvious over Davenport in view of King (Pub. No. 2003/0071842 A1)

King explicitly teaches a graphical program; that is, King teaches a program that is built graphically, using graphical representations of processing elements, etc. In this context, paragraph [0119] of King—as cited by the Office Action—teaches dragging and dropping a user-defined node into a graphical block diagram, and then allowing the user to define what the

node does (user-defined events). This type of diagrammatic programming is specific to the graphical program construction context of King, and not obviously relevant to the peripheral input system of Davenport, which is identified as being useful for translating input events for game consoles, PCs, and various other digital systems.

The Office Action fails to appreciate or even acknowledge that King's teachings are specific to the graphical program construction and execution context of King, and the Office Action thus misstates/overstates the teachings of paragraph [0119]. More particularly, paragraph [0119] discusses processing nodes in a drag-and-drop environment, and the ability of a user to define user-defined processing events, not drag-n-drop input event processing. Further, at least regarding claim 10, Fig. 15 of King appears to show a pick-list or list selection control panel, which is not the claimed drag-n-drop environment for dropping incoming input events onto input event fields and onto translated event fields.

New claims 23 and 24 are not anticipated or made obvious by Davenport

Applicant adds new claim 23 as a dependent from claim 21, which depends in turn from claim 1. Claim 23 stipulates applying a mathematical scaling to one or more event parameters of an incoming event. This scaling limitation does not appear to be taught, and Applicant notes that "multiplying" key strokes to turn one keystroke into multiple keystrokes cannot be considered a mathematical scaling of an input parameter. Support for claim 23 is found in the application as filed at, for example, paragraphs [0010], [0017], and [0020] (attenuating).

Further, new claim 24 separately breaks out the swallowing limitations of original claim 17. Thus, new claim 24 depends from independent claim 14 and stipulates that the program instructions to perform desired input event translation comprise program instructions to swallow the input event, thereby hiding it from other input event translators or hiding it from one or more other computer processes (e.g., PC operating system processing). Davenport does not teach or

suggest event swallowing and therefore does not stand as an anticipating or obviating reference for claim 24.

Closing

Applicant believes that, in view of the amendments and arguments made herein, all pending claims stand in condition for allowance. As such, Applicant looks forward to the examiner's next correspondence. Of course, the undersigned attorney encourages the examiner to call if any issues remain unresolved, or if the examiner believes that a telephone call would be productive in expediting this case toward allowance.

Respectfully submitted,

COATS & BENNETT, P.L.L.C.

Dated: 8 April 2008

Michael D. Murphy
Registration No.: 44,958

1400 Crescent Green, Suite 300
Cary, NC 27518
Telephone: (919) 854-1844
Facsimile: (919) 854-2084